MORRISON & FOERSTER LLP
MICHAEL A. JACOBS (Bar No. 111664)
mjacobs@mofo.com
MARC DAVID PETERS (Bar No. 211725)
mdpeters@mofo.com
DANIEL P. MUINO (Bar No. 209624)
dmuino@mofo.com
755 Page Mill Road, Palo Alto, CA 94304-1018
Telephone: (650) 813-5600 / Facsimile: (650) 494-0792

BOIES, SCHILLER & FLEXNER LLP
DAVID BOIES (Admitted *Pro Hac Vice*)
dboies@bsfllp.com
333 Main Street, Armonk, NY 10504
Telephone: (914) 749-8200 / Facsimile: (914) 749-8300
STEVEN C. HOLTZMAN (Bar No. 144177)
sholtzman@bsfllp.com
1999 Harrison St., Suite 900, Oakland, CA 94612
Telephone: (510) 874-1000 / Facsimile: (510) 874-1460

ORACLE CORPORATION
DORIAN DALEY (Bar No. 129049)
dorian.daley@oracle.com
DEBORAH K. MILLER (Bar No. 95527)
deborah.miller@oracle.com
MATTHEW M. SARBORARIA (Bar No. 211600)
matthew.sarboraria@oracle.com
500 Oracle Parkway, Redwood City, CA 94065
Telephone: (650) 506-5200 / Facsimile: (650) 506-7114

Attorneys for Plaintiff
ORACLE AMERICA, INC.

UNITED STATES DISTRICT COURT

NORTHERN DISTRICT OF CALIFORNIA

SAN FRANCISCO DIVISION

ORACLE AMERICA, INC.

Plaintiff,

v.

GOOGLE INC.

Defendant.

Case No. CV 10-03561 WHA

ORACLE AMERICA, INC.'S OPPOSITION TO GOOGLE INC.'S MOTION FOR JUDGMENT AS A MATTER OF LAW ON SECTIONS OF COUNT VIII OF ORACLE'S AMENDED COMPLAINT

Dept.: Courtroom 8, 19th Floor
Judge: Honorable William H. Alsup

I. INTRODUCTION

Google's motion fails to establish that it is entitled to judgment as a matter of law on any of the issues on which it has moved. Google ignores evidence in the record that contradicts its claims, on which the jury can rely to find in Oracle's favor. Google also misconstrues the law on key points, such as Oracle's entitlement to copyright protection of a collection of names that are sufficiently numerous and original, like the names of the thousands of Java API elements at issue here.

Oracle addresses each of Google's arguments below.

II. ARGUMENT

A. Google's Implementation Of The 37 Java API Packages In Android Is A Derivative Work Of Oracle's Specifications

Google's argument that the implementing source code in Android copies only unprotectable ideas from Oracle's specifications misconstrues Oracle's derivative works claim. The Android source code not only implements the functions specified in the English descriptions, it also *incorporates the structure, sequence, and organization* ("SSO") of the 37 Java API packages as described in the documentation. By copying the SSO *and* implementing the prose descriptions of the Java API specifications from English text into Java-language code, Google has created a derivative work.

Oracle's derivative works claim is well-supported by Ninth Circuit law. In *Micro Star v. Formgen, Inc.*, 154 F.3d 1107, 1112 (9th Cir. 1998), the seller of new levels for a video game claimed it had not copied any protectable expression from the game's creator because its level files "reference the source art library, but do not actually contain any art files themselves." *Id.* at 1112. The Ninth Circuit disagreed, finding that "[i]n making this argument, Micro Star misconstrues the protected work. The work that Micro Star infringes is the D/N-3D story itself." *Id.*

Similarly in the present case, Professor Mitchell testified that the narrative in the Oracle API specifications is reflected in the Android source code: "[T]he narrative is reflected in the source code because the source code is a program that in a sense carries out that narrative, does

what the explanation requires for this method." (RT at 1253:16-18 (Mitchell).) Google's expert, Dr. Astrachan, acknowledged that the Android source code was "based on the specification." (RT at 2219:7-18 (Astrachan).) That admission tracks the very definition of a "derivative work" in the Copyright Act: "A 'derivative work' is a work *based upon* one or more preexisting works . . . ." 17 U.S.C. § 101. By copying both the SSO and implementing the described functions, Google has created a work derived from the 37 Java API package documentation.

Oracle's position on this claim has been consistent. *See, e.g.*, ECF No. 396, Oracle Opp'n to Opp'n to Google MSJ at 6:

> ***Android's Source Code Is Derived From The Copied API Specifications***. Google also implemented the copied API specifications into Android source code. Google tries to downplay the significance of this, acknowledging only that it copied the "names of packages and methods and definitions." (Google Mot. at 16:15-17.) In fact, Google copied the entire hierarchical and organizational structure of the APIs into the Android source code.

Even if Google were correct that the steps a single method description requires—like isolated plot elements—are an uncopyrightable idea, Google implemented Oracle's descriptions thousands of times in the same structure, sequence, and organization in which Oracle wrote them. "[A] claim of copyright infringement can be based on infringement of a combination of unprotected elements." *Dream Games of Ariz., Inc. v. PC Onsite*, 561 F.3d 983, 988 (9th Cir. 2009). Oracle has presented substantial evidence that Google derived the structure, sequence, and organization, as well as the meaning, of the code for the Android core libraries from Oracle's API specifications. Google's expert Professor Astrachan testified that corresponding elements in Java and Android APIs are structured, sequenced, and organized in the same way:

> Q. It has to be in the same position in the application programming interface structure, sequence and organization, correct?
> A. That is correct.

(RT at 2215:24-2216:2 (Astrachan).) He further testified that the method declarations in the Android code are like the detailed headings in an outline:

> Q. And the method declarations are like the sub-sub-sub-chapter headings in this structure, sequence and organization; correct, sir?

A. I think that's one analogy that's reasonable.

(RT at 2215:2-5 (Astrachan).) Google copied Oracle's detailed outline into the Android code.

Google's argument regarding functional requirements for compatibility fails for two reasons. First, Google gets the compatibility argument in *Sega Enters. Ltd. v. Accolade, Inc.*, 977 F.2d 1510 (9th Cir. 1992) wrong. The question there was whether a party needed to copy to achieve compatibility. In *Sega,* the only way for defendant's programs to run was to copy the manufacturer's functional interface. *See, e.g., Sega*, 977 F.2d at 1515 (need to input four letter initialization code). Here, Google could have used the Java language without copying Oracle's APIs. (RT at 2212:19-2213:16 (Astrachan).) With the exception of a very few classes, the Java APIs are not required to use Java at all. (RT at 684:16-685:2 (Reinhold).) Google could have written its own APIs that provided similar functionality, as Professor Astrachan testified: "it would be possible to provide an API that performed similar functionality; not maybe exactly the same but similar." (RT at 2213:8-10 (Astrachan).) Google chose to derive its libraries from Oracle's API specifications instead.

Second, Android is not, in fact, compatible with Java. (*See* TX 383 at 8 ("Q49. Is Android Java compatible? / A. No."). As Google engineer Dan Bornstein testified, Google did not even attempt full compatibility with the Java platform:

> Q. Did Android implement all the API packages present in any particular Java Platform?
> A. No.
> Q. All right. And why not?
> A. That wasn't a goal of the project. The goal of the project was to provide something that was familiar to developers. It wasn't to provide any particular preexisting set of packages.

(RT at 1783:15-22 (Bornstein).) True compatibility with Java would have required Google to incorporate all of the Java API packages, not just some. (RT at 374:4-24 (Kurian).) Accordingly, Google cannot use compatibility to excuse its copying of a select set of Java API packages into Android.

B. The Names And Declarations Are Protectable As Part Of The Structure, Sequence, And Organization Of The 37 Java API Packages

Google moves for judgment as a matter of law that its use of both the names and declarations from the 37 Java API packages is not copyright infringement. Based on the Court's

prior ruling that individual names are not protectable under the "words and short phrases" doctrine, Google seeks to extend that to declarations as well. Google's motion should be denied for at least three reasons.

First, as explained in Oracle's comments regarding the Court's proposed jury instructions (ECF No. 997 at 4-6) and discussed at the April 27, 2012 charging conference, while individual names and short phrases (including short declarations) are not protectable on a stand-alone basis, they are protectable as part of the structure, sequence, and organization ("SSO") of the 37 API packages. (RT at 2380:14-2381:25.) Oracle is not claiming that Google infringed its copyrights by copying any single name or declaration, but by copying the SSO of the 37 Java API packages *which includes the names and declarations*. On this point, the Court indicated that it would give the following jury instruction:

> While individual names are not protectable on a stand-alone basis, names must necessarily be used as part of the SSO, and are to that extent protected by copyright.

This instruction accurately tracks Oracle's copyright infringement claim and properly recites the law on "words and short phrases." (*See* ECF No. 433 at 7-8 (Court's prior ruling that "[c]opyright protection for the selection and arrangement of elements within a work is a separate question from whether the elements themselves are protected by copyright").) *Lamps Plus, Inc. v. Seattle Lighting Fixture Co.*, 345 F.3d 1140, 1147 (9th Cir. 2003) ("a combination of unprotectable elements is eligible for copyright protection only if those elements are numerous enough and their selection and arrangement original enough that their combination constitutes an original work of authorship"); *Merch. Transaction Sys. v. Nelcela, Inc.*, No. CV 02-1954-PHX-MHM, 2009 U.S. Dist. LEXIS 25663, at *49 (D. Ariz. Mar. 17, 2009) ("the Court cannot conclude that no reasonable juror could not find creativity in the selection and arrangement of the Lexcel software's field names, let alone the remaining allegedly similar non-literal elements of the Lexcel software, sufficient to render the compilation original enough for protection").

As the Court recognized at the charging conference, the SSO of the 37 API packages is manifested in a hierarchy of named packages, classes, methods, interfaces, and fields. (RT at 2381:9-12 ("The Court: You've got to use symbols in order to have an SSO. It won't work

otherwise.").) The SSO also includes the method and class declarations defining those elements. (RT at 604:16-606:4 (Reinhold).) As illustrated below, the named elements and declarations express the SSO. By asking the Court to rule that the declarations are unprotectable, Google is trying to get an indirect ruling that the SSO is unprotectable as well. Compare the following hierarchy from the "java.nio.channels" package –

java.nio.channels
- **Object**
  - **Channels**
    - Method: newInputStream
      - Declaration: *public static InputStream newInputStream (ReadableByteChannel ch)*
  - **FileLock**
  - **Pipe**
  - **SelectionKey**
  - **Selector**
  - **AbstractInterruptibleChannel**
    - FileChannel
    - SelectableChannel
      - AbstractSelectableChannel
        - DatagramChannel
        - ServerSocketChannel
        - SocketChannel

– to the same hierarchy with the named elements and declarations removed:

____.___._____
- **Object**
  - _____
    - Method: _____
      - Declaration: _____ _____
  - _____
  - ____
  - _____
  - _____
  - _____
    - _____
    - _____
      - _____
        - _____
        - _____

- _____

(RT at 594:2-596:22 (Reinhold); TX 1046 at slides 9, 14, 15 (pp. 11, 16, 17 of 24).)

Second, while *short* declarations for methods or classes may not be protectable on a stand-alone basis, not all of the declarations within the 37 API packages are short. Google points to the simple declaration from the "max" method that Dr. Bloch discussed in his testimony (e.g., "public static int max (int arg1, int arg2)"). (RT at 786:1-787:8 (Bloch).) Many other method and class declarations within the 37 API packages are far longer and more complex. Below are some exemplary longer declarations from the source code:

- In java.security.cert.Certificate (method declaration):

  public abstract void verify(PublicKey key, String sigProvider)
  throws CertificateException, NoSuchAlgorithmException,
  InvalidKeyException, NoSuchProviderException,
  SignatureException;

- In java.net.Authenticator (method declaration):

  public static PasswordAuthentication requestPasswordAuthentication(
  String host,
  InetAddress addr,
  int port,
  String protocol,
  String prompt,
  String scheme,
  URL url,
  RequestorType reqType) { ...

- In java.nio.channels (class declaration):

  public abstract class DatagramChannel
  extends AbstractSelectableChannel
  implements ByteChannel, ScatteringByteChannel, GatheringByteChannel {

(TX 623.) Any one of these declarations is long enough that it may be copyrightable on its own. *See Johnson Controls, Inc. v. Phoenix Control Sys., Inc.*, 886 F.2d 1173, 1175 (9th Cir. 1989) ("Source and object code, the literal components of a program, are consistently held protected by a copyright on the program"). Oracle is not asserting infringement based on Google's copying of

these declarations alone, but based on copying of the SSO of the 37 Java API packages, inclusive of these declarations. However, there is no legal basis for finding that declarations in general, some of which are quite long, are not protectable.

Finally, Google's argument that the declarations "must remain exactly the same in order to ensure compatibility with Java programs calling on those 37 API packages" is a red herring. Google chose to take the 37 Java API packages because it believed they were the most appropriate for the Android mobile platform. (RT at 1783:23-1785:6 (Bornstein).) Google did not have to take all elements of the 37 packages – Google could have designed its own APIs. (RT at 2212:25-2213:10 (Astrachan).) It chose not to take the time and incur the expense of doing so. The fact that the declarations, once incorporated in the Android platform, must remain the same to ensure that Android applications will run does not excuse Google's decision to take the API packages in the first instance without a license. Moreover, because Google chose to take only a subset of the full group of Java API packages, Android is actually *incompatible* with Java, damaging Java's "write once / run anywhere" promise. (RT at 1007:6-11 (Morrill); 1331:16-1332:2, 2287:23-2288:5 (Mitchell).).

For these reasons, Google's request for a judgment that the declarations are uncopyrightable as "short phrases" should be denied.

C. Google's Literal Copying Is Not *De Minimis*

Google bears the burden of showing that the code it copied was *de minimis* as compared to the "work as a whole." *See Merch. Transaction Sys., Inc. v. Nelcela, Inc.*, 2009 U.S. Dist. LEXIS 25663, at *61 ("Thus, Nelcela will not escape liability unless it can show that the protectable elements in the Lexcel software constitute an insignificant (quantitatively and qualitatively) portion or aspect of the Lexcel software."). But regardless of who bears the burden, Google is not entitled to judgment as a matter of law because Google's literal copying of Oracle code and comments was not *de minimis*.

"[A] use is *de minimis* only if the average audience would not recognize the appropriation." *Newton v. Diamond*, 388 F.3d 1189, 1193 (9th Cir. 2004). The extent of the copying "is measured by considering the qualitative and quantitative significance of the copied

portion in relation to the plaintiff's work as a whole." *Id. at* 1195. Therefore, even if the copied material is a "quantitatively very small part" of the work as a whole, "[t]he smallness alone is not enough by itself to avoid liability." *Mktg. Tech. Solutions, Inc. v. Medizine LLC*, No. 09 Civ. 8122 (LMM), 2010 U.S. Dist. LEXIS 50027, at *9 (S.D.N.Y. Apr. 23, 2010).

Google's argument depends on comparing the copied contents from each file to the J2SE platform in its entirety. Such a frame of reference would allow plagiarists to copy entire computer files, as Google did here, and escape liability simply by claiming what they copied came from a large body of code. The Court has already rejected this argument and has stated in its proposed instructions that the "work as a whole" is the individual file from which the code was copied. During trial, Professor Mitchell testified that Google literally copied qualitatively and/or quantitatively significant portions of the copyrighted works. Google presented little evidence to rebut this testimony. In fact, Google's copyright expert, Dr. Astrachan, provided no testimony about the literally copied files. As a result, a reasonable jury could find that Google's copying was not *de minimis*.

1. The "Work As A Whole" Is The Individual File From Which The Code Was Copied, Not The Entire J2SE Platform.

Google argues that its code copying should be compared to the entire Java platform because Oracle registered it as a single work. The Court rejected this argument at the charging conference and stated that it would instruct the jury to compare the compilable code Google literally copied to the compilable code in the computer file from which it was copied. (RT at 2416:10-2418:17.)

2. Google's Copying Of Eight Decompiled Files Was Not De Minimis.

Professor Mitchell explained that Google decompiled eight Java files and copied the decompiled source code into certain Android files. (RT at 1258:7-1259:15 (Mitchell).) He presented a side-by-side comparison between a decompiled version of Oracle's PolicyNodeImpl.class file (TX 896.1) and Android's PolicyNodeImpl.java file (TX 1031). (RT at 1259:16-25 (Mitchell).) He also confirmed that Android contains decompiled versions of Oracle's PolicyNodeImpl.class, AclEntryImpl.class, AclImpl.class, GroupImpl.class,

PermissionImpl.class, PrincipalImpl.class, AclEnumeratorImpl.class. (RT at 1259:16-25; 1260:5-18.)

Once put into the proper frame of reference for comparison (file-to-file), it is clear that Google's copying of the eight decompiled files was not de minimis. Google copied each entire file, so by definition, the copied code is both quantitatively and qualitatively significant to the file it came from. No reasonable jury could find otherwise.

Google did not challenge this testimony at the trial. Although Oracle was not required to prove anything further, the decompiled code is also qualitatively significant beyond its significance to the file. The eight Oracle files concern the security governing access to files in a network. Dr. Mitchell testified that "They have to do with access control lists, which are a standard mechanism in computer security to govern access to a file or a network or other resource." (RT 1329:22-1330:5.) Google did not challenge this testimony. While Google has claimed in the past that these files were simply "test files" for Google, Google did not present evidence at trial that these were test files except to note that the file paths for these files contain the word "test." (RT 1319:1-3.) Dr. Mitchell testified that they are not test files as used by Oracle. (RT 1330:6-11.) In addition, the possibility that the corresponding Android files might be test files in no way decreases their value. Companies invest significant time and money into testing because it is important to ship their devices bug-free and thus these test files could have had a "big value" for Google as well. (RT 1330:6-1331:5.)

3. Google's Copying Of The RangeCheck Method Was Not *De Minimis*.

Professor Mitchell testified that the rangeCheck method is qualitatively significant and "useful" to Android. (RT at 1316:17-19.) He explained that the rangeCheck method operates on Android devices, including on Samsung phones. (RT at 1255:22-25, 1264:19-23 (Mitchell).) To determine how useful the rangeCheck method is on devices, Professor Mitchell experimented with an Android device and found that it called the rangeCheck method no less than 2,600 times during start up:

> Q. Did you conduct an analysis of the significance of rangeCheck to other code in the same class file?

> A. Yes.
> Q. What did you conclude?
> A. I found a number of other source code in other files that called that function. And, also, I did an experiment with the phone source code instrumented, and counted the number of times that rangeCheck was called in starting up the phone. And I found that it's called 2600 times just in powering on the device or starting the emulator.

(RT 1329:9-21.) Google did not present any testimony to rebut this.

Google selectively quotes from Professor Mitchell's testimony to try to trivialize the rangeCheck method. (ECF No. 984 at 5 ("Dr. Mitchell conceded that 'a good high school programmer' could write rangeCheck.").) But Professor Mitchell actually testified that "a good high school programmer or graduate student, *if told exactly what was needed*, could write the code." (RT at 1316:24-25 (emphasis added).) The "code has some subtlety" and "the interesting part is figuring out exactly what you wanted the function to do, more than realizing that function in Java code once that's understood." (RT at 1317:1-5.) The method might seem "[v]ery, very simple" to Dr. Bloch, who originally wrote the code while he was working at Sun, but other programmers might not find the method so simple to write. (RT 815:13-16; *see also* 755:6-8.)

4. Google's Copying Of Comments Was Not *De Minimis*

Professor Mitchell compared Oracle's CodeSource.java file (TX 623.9) against Android's CodeSourceTest.java file (TX 1039) and concluded that except for some HTML commands, certain English-language comments are "syntactically . . . identical." (RT 1262:13-1263:4.) He also testified that Android's CollectionCertStoreParametersTest.java (TX 1040) contained the "same kind of comment copying" from Oracle's CollectionCertStoreParameters.java (TX 623.10). (RT 1253:9-10.)

Comments in computer programs are protected from copying under copyright law because they are expressive. *See, e.g., Brocade Commc'ns Sys. v. A10 Networks,, Inc.*, No. 10-CV-03428-LHK, 2011 U.S. Dist. LEXIS 91384, at *7-8 (N.D. Cal. Aug. 16, 2011) (because comments are "the equivalent of explanatory asides, they are likely capable of being expressed in many different ways and therefore may be protectable expression"); *Szen Corp. v. Anderson*, No. CV-06-5073-FVS, 2007 U.S. Dist. LEXIS 42716, at *4 (E.D. Wash. June 13, 2007) (noting "the absence of precedent indicating that names and programmer comments in source code are not protected

elements"). The literally copied comments are quantitatively significant. They amount to about 25% of Oracle's CollectionCertStoreParameters.java file (31 lines out of 124 lines in the file). (TX 623.10.) They also amount to about 2.90% of Oracle's CodeSource.java file (18 lines out of 621 lines in the file). (TX 623.9.) The literally copied comments are also qualitatively significant, because they "give some guidance to programmers reading source code" and help them understand the code. (RT at 1317:21-25).

Google presented no evidence to rebut the showing that the comments are both qualitatively and quantitatively significant in relation to the individual source code files, and is therefore not entitled to judgment as a matter of law.

D. The Android Documentation Infringes Oracle's Copyrights

Google asks the court to dismiss Oracle's allegations that the Android documentation copies both the English-language descriptions and the SSO from Oracle's Java documentation. As Oracle has presented substantial evidence showing both types of copying in Android documentation, judgment as a matter of law is unwarranted.

1. The Evidence Shows That Android's English-Language Descriptions Were Copied From The Java API Specifications.

Google suggests that Oracle's evidence of English-description copying is limited to three examples. Not so. Android developer Bob Lee testified regarding three examples shown to him, but he acknowledged that the same level of similarity shown in those examples exists across the complete documentation for the 37 Java API packages within Android. (RT at 1175:25-1176:3 (Lee).) Mr. Lee admitted that the English-descriptions in Android were re-written from Sun's documentation and were therefore "substantially similar." (RT at 1191:4-13.) Indeed, he even expressed regret that such re-writing had occurred:

> I actually wasn't even a big fan of including these. I would have preferred that we just point people to Sun's site for this specific documentation because you shouldn't really be rewriting a contract. And in doing so they are going to be substantially similar.

(*Id.*) Mr. Lee's testimony alone is sufficient evidentiary basis for the jury to conclude that Google had substantially copied the English-descriptions into the Android documentation.

The full Android and Java documentation were admitted into the record, so the jury will be able to compare the two. (TX 610.2, 767.) Dr. Mitchell testified to the process he used to compare the full set of specifications for both the Java and Android APIs:

> Because the APIs in both cases are available on the web, it's fairly straightforward to open two web browsers side by side and navigate through the two APIs and libraries simultaneously in the same way and compare the way one looks on the screen with the other.
>
> By and large, that – they are really identical. They are displayed in different colors and so on. But the content is the same. You see the same classes in the same hierarchy in the same – in packages of the same name supporting the same interfaces and arranged in – in the same way.

(RT at 1244:17-1246:3 (Mitchell).) Contrary to Google's suggestion, Professor Mitchell did not merely say that the two sets of documents expressed the same idea – he testified that he compared "the way one looks on the screen with the other" and concluded they are largely identical. (*Id.*) Professor Mitchell further testified: "I don't think there's any way that a separate team could have come up with so many things that are identical except by copying the original API. There are just thousands of things that match and I don't see how that could have happened in any other way than by copying." (RT 1249:18-25 (Mitchell).) This testimony, taken together with Mr. Lee's testimony, supports Oracle's position that the English-descriptions were copied.

With respect to the three specific examples used with Mr. Lee, Google argues that no reasonable jury could find these are substantially similar. But this is a factual question ideally-suited for jury determination. *See Apple Computer v. Microsoft Corp.*, 35 F.3d 1435, 1446 (9th Cir. Cal. 1994). It requires no special legal or technical skill to compare the English-descriptions and determine if they are substantially similar. With Mr. Lee's testimony that the English-descriptions in the Android documentation were re-written from the Java documentation, a reasonable jury could conclude that the descriptions were copied. The three specific examples are reproduced below (*comparing* TX 610.2 *with* TX 767):

J2SE 5.0	**Android**

This class provides the functionality of a cryptographic Cipher for encryption and decryption. In order to create a Cipher object, the application calls the Cipher's get-instance method, and passes the name of the requested transformation to it. … When requesting a block cipher in stream cipher mode (e.g., DES in CFB or OFB mode), the user may optionally specify the number of bits to be processed at a time, by appending this number to the mode name as shown in the "DES/CFB8/NoPadding" and "DES/OFB32/PKCS5Padding" transformations. If no such number is specified, a provider-specific default is used	This class provides access to implementations of cryptographic ciphers for encryption and decryption. Cipher classes can not be instantiated directly, one has to call the Cipher's getInstance method with the name of a requested transformation, optionally with a provider. ... When a block cipher is requested in stream cipher mode, the number of bits to be processed at a time can be optionally specified by appending it to the mode name. e.g. "AES/CFB8/NoPadding". If no number is specified, a provider specific default value is used.

J2SE 5.0	**Android**
A pair of channels that implements a unidirectional pipe. A pipe consists of a pair of channels: A writable sink channel and a readable source channel. Once some bytes are written to the sink channel they can be read from source channel in exactly the order in which they were written.	A pipe contains two channels, forming a unidirectional pipe. One is the writable sink channel, and the other is the readable source channel. When bytes are written into the writable channel they can be read from the readable channel. Bytes are read in the order in which they were written.

J2SE 5.0	**Android**
A CipherInputStream is composed of an InputStream and a Cipher so that read() methods return data that are read in from the underlying InputStream but have been additionally processed by the Cipher. The Cipher must be fully initialized before being used by a CipherInputStream.	This class wraps an InputStream and a cipher so that read() methods return data that are read from the underlying InputStream and processed by the cipher. The cipher must be initialized for the requested operation before being used by a CipherInputStream.

Oracle has presented sufficient evidence for the jury to determine that the English-descriptions in the Android documentation were copied from the Java documentation.

Accordingly, Google cannot be entitled to judgment as a matter of law on this issue.

2. Evidence Also Shows That Google Copied The SSO Of The Java API Packages From Java Documentation Into Android Documentation

Google admits that the SSO of the 37 Java API packages as expressed in the Android documentation is identical to the SSO expressed in the Java documentation. (ECF 984 at 10.) Google argues that Oracle's claim for SSO copying into the Android documentation should be dismissed because it purportedly overlaps with Oracle's claim for SSO copying into the Android code. But Google is liable for copying the SSO into the Android documentation as well and Oracle is entitled to a judgment on that claim.

There is no question that Google has copied the SSO of the 37 Java API packages into the Android documentation. As explained by Dr. Reinhold, the structure expressed in the API documentation is the same as the structure within the compilable code because the code is run through the Java Documentation Extractor (or "Javadoc") to pull out the structure and English language comments, and produce a webpage that reflects the same SSO of the API that is in the code. (RT at 606:14-608:3 (Reinhold); TX 1046 at p.19 of 24.) The Android documentation is created the same way from the Android code. (RT at 1169:8-15 (Lee agreeing that like Java documentation, Android documentation was "created by a tool that actually reads portions of the source code and then places it in a kind of template that's available on the web as a source of documentation.")) It is undisputed that the SSO of the 37 Java API packages is found, identically, in both the Java and Android code and documentation. (ECF 984 at 10 (Google's JMOL motion: "As such, if based on the same starting point – the names of the 37 API packages at issue – the structure, sequence, and organization of the Android and Java documentation inevitably will be the same").)

Regardless of whether the SSO is expressed in the compilable code or the API documentation, it is protectable expression in both cases. In *Am. Dental Ass'n v. Delta Dental Plans Ass'n*, Judge Easterbrook found that the structure of dental code which organized various dental procedures into a hierarchy represented in three formats – numerically, by short description, and by long description – was equally protectable regardless of which format was

copied. 126 F.3d 977, 979, 980-981 (7th Cir. 1997) ("The long description is part of the copyrighted work, and original long descriptions make the work as a whole copyrightable. But we think that even the short description and the number are original works of authorship."). The infringer copied "most of the numbering system and short descriptions from the ADA's Code," and the Court did not distinguish one format from the other. It explained that taxonomies, such as the West Key Number System and the dental code, are not "systems" and are protectable as expression. 126 F.3d at 978 (7th Cir. 1997) (noting that "[b]lueprints for large buildings (more committee work), instruction manuals for repairing automobiles, used car value guides, dictionaries, encyclopedias, maps" are protectable original expression). Analogously, if the SSO of the 37 Java API packages is protectable as expressed in source code, it is also protectable as expressed in the API documentation. Far less creative structures have been found subject to copyright protection in this Circuit. Courts in this Circuit have repeatedly extended copyright protection to structure expressed in written documentation. *See, e.g., CDN Inc. v. Kapes,* 197 F.3d 1256, 1262 (9th Cir. 1999) (prices in guide for collectible coins); *Practice Mmgt. Info. Corp. v. Am. Med. Ass'n,* 877 F. Supp. 1386, 1390-92 (C.D. Cal. 1994), *aff'd in relevant part*, 121 F.3d 516 (9th Cir. 1997) (numerical codes for medical procedures); *Jacobsen v. Katzer*, 2009 U.S. Dist. LEXIS 115204, at *9-10 (N.D. Cal. Dec. 10, 2009) (text files reflecting decoder information from model railroad manufacturers).

The evidence shows that Google copied the SSO of the 37 Java API packages into both the Android compilable code and documentation. Oracle is entitled to a judgment on both forms of copying. Google's motion for judgment as a matter of law on this point should be denied.

E. Google Incorrectly Claims That Oracle Failed To Prove The Contents Of The Copyrighted Works.

Google also raises the hyper-technical challenge that Oracle "has not offered any proof of the actual contents of the works that are the subject of its copyright registrations." (ECF No. 984 at 10-11). Google's challenge is baseless.

First, Oracle submitted the copyright registrations, which claim on their face to cover "Java 2 Standard Edition 1.4" and "Java 2 Standard Edition, Version 5.0" (TX 464, 475, 476),

and authenticated and submitted into evidence the full contents of both these versions of the platform. (TR 689:21-691:23, TX 622, 623.) Oracle is entitled to a presumption that the facts stated on the front of the copyright registration are correct. Section 410(c) of the Copyright Act provides that the certificate of registration "shall constitute prima facie evidence of the validity of the copyright and of the facts stated in the certificate." 17 U.S.C. § 410(c). Google accordingly "has the burden of rebutting the facts set forth in the copyright certificate." *United Fabrics International, Inc. v. C&J Wear, Inc.*, 630 F.3d 1255, 1257 (9th Cir. 2011). Google has submitted no evidence whatsoever showing that the registrations cover anything other than what they purport to claim, or that the registered code differs from what was identified at trial.

Second, because Google insisted on pursuing this issue, Oracle submitted additional evidence of the content of the registrations. Chief Java Architect Mark Reinhold testified that the written source code deposit that accompanied the certified copy of the registration for J2 SE version 1.4 — which he wrote — does in fact correspond to the source code for J2 SE version 1.4. (TR 2233:18-2234:19, TX 3530). He similarly testified that the written source code deposit included with the certified copy of the registration for J2 SE version 5.0 represented part of the source code for version 5.0, and that the CD-ROM entered into evidence as Exhibit 1076 is a copy of the binary code for version 5.0 along with the documentation and other tools. (TR 2234:20-2238:19, TX 3529). The loop on Exhibit 1076 was closed when Oracle attorney Tiki Dare authenticated the transmittal letter and accompanying documentation that was transmitted to the Copyright Office along with Exhibit 1076 when registering version 5.0. (TR 2257:5-2267:2, TX 1077, 1078, 1081). Again, Google has not submitted any evidence showing that this evidence is incorrect, and certainly is not entitled to judgment as a matter of law.

F. Google's Challenge To Authorship Was Predicated On The Court's Acceptance Of The Copyrighted Works As "Collective Works" And Is Now Moot

Google's motion argued that the copyrighted works should not be considered to be collective works and, as an alternative argument, raised a challenge to authorship in the event the works were deemed to be collective works:

> In sum, if the Court accepts Oracle's "collective work" argument, then Google is entitled to judgment as a matter of law of non-infringement of each of the component parts of the registered works, because Oracle has not proved authorship of the constituent elements.

(ECF No. 984 at 12. *See also id.* at 11 ("If, however, the Court accepts Oracle's 'collective work' argument, then Google is entitled to judgment as a matter of law of non-infringement of each of the component parts of the registered works, because Oracle has not proved authorship of the constitutent elements.")

Oracle has withdrawn the characterization of the registered works as "collective works," in part in response to Google's complaint. (*See* TR 2134:11-17, 2134:7-11). Accordingly this argument is now moot.

III. CONCLUSION

Google has not shown that it is entitled to judgment as a matter of law on any of the issues on which it has moved. For all the above reasons, Google's motion should be denied.

Dated: April 29, 2012

MICHAEL A. JACOBS
MARC DAVID PETERS
DANIEL P. MUINO
MORRISON & FOERSTER LLP

By: /s/ *Michael A. Jacobs*
Michael A. Jacobs

Attorneys for Plaintiff
ORACLE AMERICA, INC.